CLAIMS:

We claim:

1. A method for managing an interposed reverse proxy comprising the steps of:

comparing within a markup language document, a host address for said markup language document and a codebase address for a code base supporting logic disposed within said markup language document; and,

if said host address and said codebase address differ, concluding that a reverse proxy has obscured from view a server source of said markup language document.

2. The method of claim 1, further comprising the steps of:

retrieving a server affinity identifier for said server source from configuration tags for said logic; and,

responsive to said conclusion, attempting a tunneled connection to said server source through said reverse proxy by inserting said server affinity identifier in an address specified in said attempt.

3. The method of claim 1, wherein said comparing step comprises the steps of:

retrieving said markup language document for rendering within a content browser;

parsing said markup language document to identify logic embedded within said markup language document;

locating within said logic, a tag denoting a host address for said markup language document as viewed by a server which generated said markup language document;

further locating within said logic, another tag denoting an address for a code base for said logic; and,

comparing said addresses to determine if said host address differs from said address for said code base.

4. The method of claim 2, wherein said retrieving step further comprises the step of locating said server affinity identifier within a tag disposed within said logic.

5. The method of claim 2, wherein said attempting step comprises the steps of:

combining an address for said reverse proxy with said server affinity identifier and a string specifying a particular desired resource within said server source;

forming a hypertext transfer protocol (HTTP) compliant request using said combined address;

encapsulating non-HTTP data within said HTTP compliant request; and,

forwarding said HTTP compliant request to said reverse proxy.

6. A system for managing a reverse proxy interposed between a client and server, the system comprising:

detection logic disposed within the client and programmed to detect the interposed reverse proxy by comparing host and codebase addresses embedded within content provided by the server; and,

simulation logic further disposed within the client, said simulation logic being responsive to said detection logic and programmed to selectively incorporate a server affinity identifier in requests addressed to the interposed reverse proxy to ensure re-routing to the server.

7. The system of claim 6, wherein said detection and simulation logic are embodied in an applet executing within a virtual machine disposed within the client.

8. The system of claim 6, wherein said simulation logic comprises hypertext transfer protocol (HTTP) tunneling logic for establishing a tunneled connection through the reverse proxy to the server.

9. The system of claim 6, wherein said content comprises at least one address for additional resources provided by the server, said at least one address comprising a relative address and not an absolute address.

10. The system of claim 6, further comprising a user-selectable toggle embedded within said content for deactivating and reactivating said detection and simulation logic.

11. A machine readable storage having stored thereon a computer program for managing an interposed reverse proxy, the computer program comprising a routine set of instructions for causing the machine to perform the steps of:

comparing within a markup language document, a host address for said markup language document and a codebase address for a code base supporting logic disposed within said markup language document; and,

if said host address and said codebase address differ, concluding that a reverse proxy has obscured from view a server source of said markup language document.

12. The machine readable storage of claim 11, further comprising the steps of:

retrieving a server affinity identifier for said server source from configuration tags for said logic; and,

responsive to said conclusion, attempting a tunneled connection to said server source through said reverse proxy by inserting said server affinity identifier in an address specified in said attempt.

13. The machine readable storage of claim 11, wherein said comparing step comprises the steps of:

retrieving said markup language document for rendering within a content browser;

parsing said markup language document to identify logic embedded within said markup language document;

locating within said logic, a tag denoting a host address for said markup language document as viewed by a server which generated said markup language document;

further locating within said logic, another tag denoting an address for a code base for said logic; and,

comparing said addresses to determine if said host address differs from said address for said code base.

14. The machine readable storage of claim 12, wherein said retrieving step further comprises the step of locating said server affinity identifier within a tag disposed within said logic.

15. The machine readable storage of claim 12, wherein said attempting step comprises the steps of:

combining an address for said reverse proxy with said server affinity identifier and a string specifying a particular desired resource within said server source;

forming a hypertext transfer protocol (HTTP) compliant request using said combined address;

encapsulating non-HTTP data within said HTTP compliant request; and,

forwarding said HTTP compliant request to said reverse proxy.